SMPTE Journal 2015

Taking the Pixel out of the Picture

By Philip Willis, John Patterson, Peter Balch, and Jan Paxton

We have developed a contour-based image and movie representation that takes the pixel out of the picture. We use contours, which are scale-free and can readily be rendered back to an image at a new resolution independent of the original. They are easy to rotate and zoom by fractional amounts. They can act as a universal intermediate during post-production. They provide a single delivery mechanism, whether to mobile phone, television screen, or cinema. They are future-proof, taking high-definition resolution and beyond in their stride. This is not a disruptive technology: Moving between pixels and contours can happen at any stage in the pipeline and does not need special cameras or displays.

INTRODUCTION

Photography, whether still or movie, is now dominated by pixel representations. Cameras capture pixels, and displays show them. In between, there are digital production pipelines for editing, for visual effects, grading, and many other operations. Pixels are data-hungry, so there are compressors and decompressors for the delivered product. To support the delivery, standards have evolved to ensure that the consumer market can be addressed, whether via digital video disc (DVD), theater, download, or satellite. Pixels are not going to go away any time soon.

They are not without their problems. To create samples of the optical image in-camera is to fix

the spatial resolution at the limit of the sensor. To rotate pixel images by small amounts, such as to correct for camera tilt, is computationally demanding. So too is changing the resolution when combining one image with another or when preparing a movie for a variety of output devices of differing resolution.

These operations are found in the processing pipeline of any post-production, visual effects company. The image capture and image display have to be pixel-based, but this processing pipeline does not.

We have been experimenting with contourized images, in which we extract contours of constant brightness from the pixel original. The aim is to replace discrete pixels with a continuous, resolution-independent representation, for which contours are one solution. Contours are smooth curves passing through the pixels. A given pixel can have several contours passing through it, if the brightness is changing very rapidly in its locality. A pixel may have no contours passing through it if the locality is unchanging. A contour is arbitrarily smooth, so it does not suffer from sampling effects. There is only one discrete choice needed: the spacing of the brightness of the contours.

The advantages of the contour form include the following.

1. The representation is resolution-independent. Of course, this does not mean that a contour image contains more detail than the pixel original, but there is nothing that prevents it from being immediately combined with other contour images.
2. For the same reason, any given contour image can be directly rendered to any desired

resolution of pixel image, to suit the intended display. Arbitrary zoom in and out is as easy as just showing the image. This makes it a universal intermediate form for delivery, with conversion to pixels only needed at the point of viewing.

3. Contours follow brightness levels; this is often where the visually important information lies and needs to be retained. This also means failure cases are more acceptable, with none of the blockiness seen in discrete encoders.

4. In stark contrast to pixels, contours extend across the image and so are more meaningful image "atoms" than the necessarily very local information in pixels. They are a more useful first step on the path to processing the image for features.

They have some disadvantages too.

1. To view the image, the gaps between the contours have to be interpolated at the desired pixel resolution. A poor choice of rendering would generate a smudged image, soft and out-of-focus in appearance. The same is true if the brightness gaps between contours are too large for the image in question.

2. A simple contourization is an alternative representation to pixels. It is not a compression technique. It does, however, open up completely new approaches to compression.

3. All existing pipelines are pixel-based.

When synthesizing an image by computer graphics (CG), a virtual model of the desired object is first constructed. This is then illuminated and rendered to produce the pixels, which make a two-dimensional (2D) picture of what is often a three-dimensional (3D) model. The model is an important component because it allows us to manipulate the object. In contrast, when we take a photograph, there is no underlying model: We only have a record of the pixels. We are suggesting that contours are a scale-free model of the optical image, one which we can

manipulate as easily as vector graphics. Moreover, as with CG, we do not need to decide the output resolution until we have chosen the viewing device and what to look at in the picture.

In what follows, we will describe some earlier related work and then our own approach. We will explain how we approach the need for high-quality results and discuss some professional test examples. Finally, we will summarize where we have moved the state of the art in continuous representations of images.

PREVIOUS RELATED WORK

The literature has focused on drawn images. When an artist draws, they move a drawing implement over the paper. Their hand movements are a continuous representation of the lines of the image. If these movements are recorded with auxiliary information about the pen, pencil, or brush, the image can be synthetically redrawn at any desired pixel resolution. In this case, the record of the movements is the model. Examples based on drawings include Orzan et al.'s[1] work. We have explored this ourselves[2–6] and have also produced professional-quality, highly styled animations[7,8] this way.

Contouring has a long history in map-making. Maps are based on point heights sampled irregularly over the landscape, generally at quite low resolution. In contrast, we are starting from pixels at the finest resolution. Moreover, cartographers fill between contours with flat color. For images, this produces a posterized result of limited value to the professional industry. Nakajima et al.[9] have explored this approach for images. Although there has been work on reconstructing 3D landscapes from such data, the accuracy is much less demanding than we need.

More recently, we have been investigating the use of contours run through pixels.[7,8] A key factor in this is the method used to reconstruct the image. We work bidirectionally, from an inner contour to the surrounding one and from the surrounding one to the inner one. The inner contour is dilated towards the outer, and the outer is contracted towards the inner. For each desired output pixel, this gives a measure of the shortest paths to its surrounding contours. In turn, this allows us to interpolate the brightness contributed to that pixel by the two contours. To render a contour image, we first choose the desired pixel resolution and the orientation of the pixel grid relative to the contours. Then, we generate each pixel from the sampling process just described. We are also able to cope with multiple contours passing through the pixel and with contours containing two or more separate contours. Our papers give details and also show some sample images.[7,8]

Since that work was reported, we have been working on a version that aims to reach full professional quality. This work has been undertaken with a grant from the UK's Technology Strategy Board (TSB), whose brief is to move university research into industrial use. Our partners in this are Root6 (the lead partner), Smoke & Mirrors, and Ovation Data Services. Root6 is a London-based provider of the technology used by post-production and special effects companies. Smoke & Mirrors is a dedicated visual effects and full-service post-production company, working for the advertising, film, and music industries. They are in London, New York, and Shanghai. Ovation Data Services is a major U.K.-based company servicing demanding applications in "big data" and visualization for the oil and gas industries, video, post, and broadcast.

Our recent work has concentrated on two aspects: fitting the software into the professional pipeline at Root6 and evaluating the quality of images and movies produced. The first of these is a purely technical matter, but it is important because it allows us to experiment beyond the university laboratory. In turn, we expect this to lead to improvements. The quality aspect is what we will discuss in this paper, with some results to date.

CREATING CONTOURS FROM PIXELS

We will give here a brief outline of how we create the contours. We start with a pixel image, but we also accept that any captured image will not be a perfect match for the optical image. No matter how good the capture device, there will still be a small amount of residual noise and also some rounding in the digital output. For professional capture devices, this will be small but not zero. As a result, there is a small degree of freedom in which the contour can run, pixel by pixel. This is useful because, as our experimental results will show later, excessively constraining the contour produces needlessly large amounts of data.

Our first step may seem counterintuitive: We double the size of the image. Put another way, each source pixel is now surrounded by eight new pixels. We call this stage *pixel-mapping*. Any conventional tool doing this will generate the eight from the original pixels grouped around the source pixel; in effect, resampling. So, the eight surrounding pixel-map pixels will not be identical, but each will vary a little according to the nearby source pixels. In effect, we have a map of the details "within" the original pixel, details which were averaged by the capture process. What we are effectively doing here is getting a measure of which way the image varies about the chosen pixel and at what rate.

To construct a contour through the grid of pixels, we use standard techniques but take advantage of the uncertainty in the pixel data due to sampling noise and rounding. There is a narrow "ribbon of uncertainty" within which the contour must lie. That is, we are modeling the resolvability of the contour within the given captured pixels. If it is too wide, then the image will be degraded, and errors will be introduced. If the ribbon width is narrow and justified by the uncertainty, this does not affect the output image quality. If it is narrower than the uncertainty justifies, then the ribbon will overly-constrain the contour. The practical effect of this is that the contour may be forced to curve or change direction more than the underlying data needs, increasing the contour storage cost dramatically. By way of analogy, a racetrack driver alone on a circuit can choose a smooth path using the full width of the road to achieve this. Drivers even refer to this as "straightening out the bends". In the presence of many other cars or a very narrow section of track, the driving line is too constrained for this to happen.

This ribbon matters because, viewed across the image area as a whole, there are many ways we could fit a closed curve within the ribbon. So we are free to use one that uses less data, as long as the resulting contour does not go outside the ribbon. If we do not do this, then one result can be many tiny noise-induced loops, each within a single pixel. We use Bézier segments to build our contours, and so we can use this to minimize the number of segments needed. More detail on this, and indeed the contour rendering process, is given in our earlier papers.[10,11] We also show numerical results for the "ribbon of uncertainty" in the Results section, which confirm that even sub-pixel ribbons (that is, as fine as can be expected) still produce relatively compact output files.

CURRENT RESULTS

Although we have not paid much attention yet to data sizes, we will give some indicative results. We have put most of our effort into comparing the quality of demanding, professionally sourced images before and after our encode/decode process, and we will also say more about that.

Given that this is the first time we have moved the technique out of the laboratory, it is worth first saying what we were trying to achieve with the UK TSB funding, which has driven the project for the last 2 years. When we started work in this area, we found very little had been done. Our first implementation gave results that exceeded our expectations by giving convincing reconstructions with both fine detail and with slowly shaded areas. We quickly came to realize that it opened a large and unexplored territory, well beyond anything that a small research team could accomplish. In particular, we had little access to professional quality video. We did have access to several major London-based effects and post-production companies, having completed a study of industry needs a short while before. We also had contacts in a wide variety of digital media companies, through our national Centre for Digital Entertainment, which works exclusively with the industry. Rather than taking it forward as a piece of pure university research, we looked for company partners who could give us realistic challenges and who knew what the wider industry was likely to value. This led to a two-year TSB-funded project, just completed, with Root6, Smoke & Mirrors, and Ovation Data Services.

The decision was to concentrate on the commercial internal pipeline and the potential for our

work to be a universal intermediate, rather than the final delivery to the consumer's screen. This also avoided the work being disruptive. Conventional codecs can be used as they are now, and indeed the internal pipeline can continue to use pixels and pixel-based tools at any stage. Contourized tools can gradually replace them as they are written, but there is no pressure to do so completely. To give one example, a company like Smoke & Mirrors routinely delivers 50 or more variants of a single advert movie. In part, this is because of regional language and cultural variants, but there are still many variants that depend on resolution. Even television requires several versions according to territory and technologies. Add in the web, mobile phones, pads, theatres, etc., and the variants multiply rapidly. With our approach it becomes possible to deliver a single file of a movie, with a small file of data specifying the rendering information needed for each target device. The host organization uses the information relevant to its needs to render out the final version, which is then delivered to the consumer.

With this emphasis on company-internal pipeline processing, we have concentrated most strongly on visual quality in every frame, which is a high bar.

Early Image-Quality Results

The first test was run with the familiar "Marcie" test image. **Figure 1** shows our rendered version. **Figure 2** shows the red-green-blue (RGB) difference image and the difference image greatly contrast-enhanced to show the issues.

With the Marcie test image and a fixed brightness interval of 8 in all three channels, we generated 16,441 Y-contours, 7908 U-contours, and 5013 V-contours. With all calculations and

output storage in double precision, times were slow: 70 sec to contourize and over 200 sec to recreate the image at only 720 ×567 pixel resolution. There is also some color bleaching overall. On the positive side, the picture is identifiable and complete, and the error image is small. For a first attempt, this was nevertheless promising, certainly better than we had seen from other work. We had intended to quantify the errors, but that was put to one side by the unexpected failure to cope with flat-shaded areas, while performing well in more subtle regions. The bar to the right is noticeably poor. Closer inspection of the RGB exaggerated differences image shows the text in that region influencing contours further away, as does the foliage at the vertical boundary. This is essentially an edge-effect: The contours inside the flat-shading are trying to bridge an undefined color just outside the image with rapidly changing colors near the text and the foliage. This quick test turned into a quick learning curve. The problems included a bug exaggerating the flat-shade problem and were easily addressed. We then started a run of quantifiable tests, coupled with visual checking by company experts. These included a range of commercial images, which we cannot include for copyright reasons, and also the test images shown here.

Data Size and Timing Results

With the code improvements after the Marcie test, we ran quantitative tests.

These tests used YUV, so we could independently choose the difference in value between adjacent contour levels in each plane. For example, we might choose increments of 8. If the raw data was just 8 bits per channel, that would give 32 contour levels.

Figure 3 (Remains of the Feast) and **Fig. 4** (Sienna Lightning) were tested. These are 1600 ×1063 original resolution. Given that most of the information is in the Y channel, we might

reasonably use a more generous spacing in U and V than in Y. Working in YUV space, we set the brightness steps between contours from (10,10,10) to (12,16,16) to (16,20,20), to see how the number of contours and the encode/decode times varied.

Table 1 shows the number of contours in each of Y, U, and V as we vary the contour brightness spacings. As we expect, this confirms we need fewer contours with larger spacing and also that most of the contours are in the Y channel.

YUV Contour Spacing	**Remains of the Feast**	**Sienna Lightning**
(10,10,10)	(33120, 2856, 1866) = 37,842	(72425, 14341, 8517) = 95,283
(12,16,16)	(27555, 1694, 1010) = 30,259	(61718, 10780, 6744) = 79,242
(16,20,20)	(21026, 1786, 852) = 23,664	(48357, 7686, 3896) = 59,939

Table 1. Number of contours for various contour spacings.

When fitting contours we can force the contours to go exactly through the calculated point (i.e., within floating point accuracy). Alternatively, we can permit some error, so that the contours are slightly less constrained, needing less data to represent them, even though the number of contours is unchanged. **Table 2** is the "zero-error" option, where we effectively assume there is no "ribbon of uncertainty." The number of contours is as before.

YUV Contour Spacing	**Remains of the Feast (encode/decode sec)**	**Sienna Lightning (encode/decode sec)**
(10,10,10)	23/27	41/68
(12,16,16)	17/21	32/59
(16,20,20)	14/19	24/43

Table 2. Encode and decode times for various contour spacings; no error allowed.

We then ran a series of tests with different ribbon widths. This had only a small effect on the encode/decode times but dramatically reduced the data size. For example, even a very small width, within a pixel position, reduced the data needed by a factor of five, from ~30 Mbytes to ~6 Mbytes (please keep in mind that this is still a file of explicit double-precision numbers, with no attempt to compress or to store in an efficient format). Increasing the permitted error further reduced the data, but now only gradually. What this suggests strongly is that the zero-error version (no ribbon) is overly constrained; that some room to flex is well worth the return; and that a good-quality version needs scarcely more data than a poor-quality one. This ribbon approach is justified by the earlier remarks on noise and rounding. Larger choices of ribbon width would introduce "real" error and be lossy, useful in some circumstances but not for our target professional-user pipeline.

FUTURE TESTING

We have performed a range of similar tests on other images, including frames from high-definition (HD) movies. We have run a first set of perceptual tests with expert evaluators from the industry. These are indicating that we achieve excellent quality; only after close scrutiny are any differences noticed, and these are not visually objectionable. We now need to take this further and work out which of the parameters can be usefully pushed to reduce the data and indeed the times to encode and decode, without perceptual loss of quality.

All our contour brightness spacings to date have been imposed by us. An additional investigation would set these spacings dynamically according to the nature of the image, or indeed vary the spacing according to high-key or low-key sections.

Similarly, we might be able to simplify the contours at render-time, where the resolution of the device is too low for this to be an issue. Contours are intrinsically hierarchical, so this might help

us do this cleanly.

Compression is always an issue. We can make obvious changes such as replacing floating point with fixed point, reducing the number of bits we write to file, and running conventional lossless compressors over the file. To be investigated still is using more efficient coding of each contour and what gain there might found contour to contour. This is new territory, and all needs exploration.

We have encoded and re-rendered professional movie sequences frame by frame. We are not able to include frames here for copyright reasons. A movie example of some demanding footage has been generated and can be viewed on YouTube™, by searching for "vector-based video codec".

CONCLUSION

We have demonstrated high-quality continuous images based on contouring pixels. Preliminary tests show promise, primarily as a standard intermediate form, but also as a good method for changing resolution and alignment. This has opened up a series of engineering and perceptual experiments that need to be completed for a full evaluation, such as a consortium might do ahead of bringing it to market. At that point, there would also need to be a range of professional tools supporting it.

Our initial aim is to support the post-production industry by achieving high quality. The use of a “universal intermediate,” which models the picture rather than storing pixels, is of value to them, but it also meets the needs of any company needing to deliver to consumers across a variety of platforms that have varying resolutions. There is now an open field of new experimentation and tool creation, including for stereo movies. There is also an opportunity to rethink compression and indeed to use existing graphics cards to speed rendering.

The push to ever higher resolutions, 4k now with more to come, is guaranteed to need square-law more raw pixels. With contours, it may be possible to put the data where they are most needed, near regions of rapid change and achieve less than square law data growth. We have made a start, but there is a lot to be explored, and it needs a concerted effort by a group of technology providers and researchers to provide strong answers on a short time frame.

ACKNOWLEDGMENTS

We are grateful to the UK's Technology Strategy Board for 24 months of funding for this project. The authors are grateful to our partner companies Root6, Smoke & Mirrors, and Ovation Data Services for contributing their time and for being such collegiate partners. Jan Paxton kindly allowed us to use the images in **Figures 3 and 4**.

REFERENCES

1. A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin, "Diffusion Curves: A Vector Representation for Smooth-Shade Images,"
Proc. ACM SIGGRAPH, 27(3), Aug. 2008.

2. P. J. Willis, "A Paint Program for the Graphic Arts in Printing," *Proc. Eurographics 84*, North Holland, Copenhagen, pp. 109–120, 1984.

3. G. W. Watters and P. J. Willis, "UltraPaint: A New Approach to a Painting System," *Computer Graphics Forum*, 6(2):125–132, May 1987.

4. P. J. Willis and G. W. Watters, "Scan Converting Extruded Lines at Ultra High Definition," *Computer Graphics Forum*, 6(2):133–140, May 1987.

5. Frédéric Labrosse and Philip Willis, "Towards Continuous Image Representations," *Proc.*

Winter School of Computer Graphics and Visualisation 2001, University of West Bohemia, pp. 206–213, Feb. 2001.

6. Max Froumentin, Frédéric Labrosse, and Philip Willis, "A Vector-based Representation for Image Warping," *Computer Graphics Forum*, 19(3): 2000; Eurographics Conference issue C385-C394 and C428, ISSN 0167-7055.

7. Fabian Di Fiore, Frank Van Reeth, Philip Willis, and John Patterson, "Highly Drawn Stylized Animation," *24th Computer Graphics International Conference*, Hangzhou, China ,Springer, 2006.

8. Fabian Di Fiore, Frank Van Reeth, John Patterson, and Philip Willis, "Highly Stylised Animation," *Visual Computer*, 24(2):105–123, Feb. 2008.

9. M. Nakajima, T. Agui, and M. Takeda, "Coding Method of Gray-Valued Image by Density Contour Lines," Tech. Rep. I.E.C.E. Japan IE83-74, pp.1–6, 1983.

10. J. W. Patterson, C.D. Taylor, and P J. Willis, "Reconstructing Vectorised Photographic Images," *CVMP 2009—The 6th European Conference for Visual Media Production*, IEEE Computer Society, London, pp. 15–24, 2009.

11. J. W. Patterson, C. D. Taylor, and P. J. Willis, "Constructing and Rendering Vectorised Photographic Images," *J. Virtual Reality and Broadcasting*, 9(3): 2012.

Figure Captions

Figure 1. "Marcie" test image: rendered.

Figure 2. "Marcie" difference and exaggerated difference images.

Figure 3. Remains of the Feast.

Figure 4. Sienna Lightning.

BIO

Philip Willis, Centre for Digital Entertainment, Department of Computer Science, University of Bath, Bath BA2 7AY, U.K.; P.J.Willis@bath.ac.uk.

John Patterson, Peter Balch, Jan Paxton, Department of Computer Science, University of Bath, Bath BA2 7AY, U.K.

Figure 1. “Marcie” test image: rendered

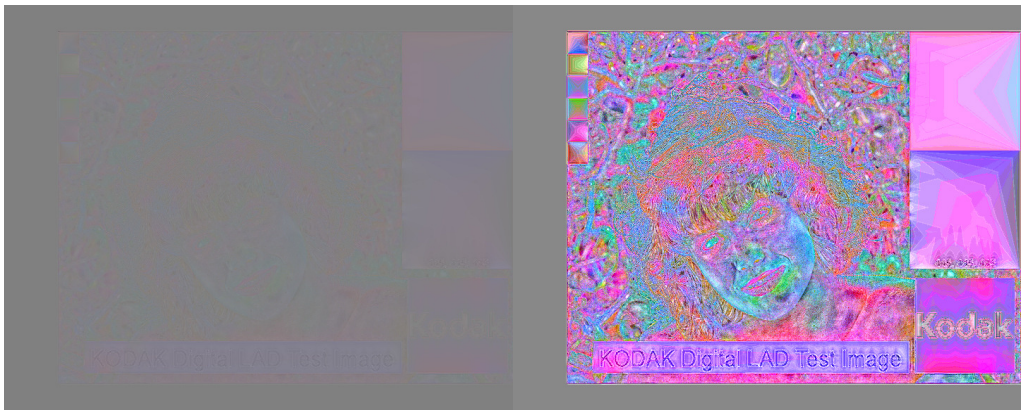

Figure 2. “Marcie” difference and exaggerated difference images

Figure 3. Remains of the Feast

Figure 4. Sienna Lightning